

Name : N.ASHWATH

Roll no : CH.SC.U4CSE24133

DAA – Quick Sort Using 3 Conditions

I) First Element as Pivot

Working :

Use Quick Sort and sort this array by using

i) First Element as a Pivot
ii) Last Element as a Pivot

i) 157 110 147 122 111 149 151 141 123 112 117 133

SWAP

STEP-1:

Pivot 157
133 110 147 122 111 149 151 141 123 112 117 157

STEP-2:

Pivot 133
133 110 117 122 111 149 151 141 123 112 147 157

STEP-3:

Pivot 133
133 110 117 122 111 112 151 141 123 149 147 157

STEP-4:

Pivot 133
133 110 117 122 111 112 123 141 151 149 147 157

SWAP

STEP-5:

Pivot 123
123 110 117 122 111 112 133 141 151 149 147 157

SWAP

STEP-6:

Pivot 112
112 110 117 122 111 123 133 141 151 149 147 157

SWAP

STEP-7:

Pivot 112
112 110 111 122 117 123 133 141 147 149 151 157

SWAP

STEP-8:

Pivot 111
111 110 112 122 117 123 133 141 147 149 151 157

SWAP

STEP-9:

110 111 112 117 122 123 133 141 147 149 151 157

It takes 9 steps to completely sort the unsorted array using first element as pivot element.

Program :

```
#include <stdio.h>

int partitionFirst(int a[], int low, int high) {
    int pivot = a[low];
    int i = low + 1, j = high, temp;

    while (i <= j) {
        while (i <= high && a[i] <= pivot)
            i++;
        while (a[j] > pivot)
            j--;
        if (i < j) {
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
        }
    }

    temp = a[low];
    a[low] = a[j];
    a[j] = temp;

    return j;
}

void quickSortFirst(int a[], int low, int high) {
    if (low < high) {
        int p = partitionFirst(a, low, high);
        quickSortFirst(a, low, p - 1);
        quickSortFirst(a, p + 1, high);
    }
}

int main() {
    int a[] = {10, 7, 8, 9, 1, 5};
    int n = 6, i;

    quickSortFirst(a, 0, n - 1);

    printf("Sorted array:\n");
    for (i = 0; i < n; i++)
        printf("%d ", a[i]);

    printf("\nCH.SC.U4CSE24133\n"); // Edited Line

    return 0;
}
```

Output :

```
Sorted array:  
1 5 7 8 9 10  
CH.SC.U4CSE24133
```

II) Last Element as Pivot

Working :

ii) Last Element as Pivot Element :-

STEP-1 157 110 147 122 111 149 151 141 123 112 117 133
l>p Pivot
SWAP

STEP-2 117 110 147 122 111 149 151 141 123 112 157 133
l<p l<p l>p Pivot
SWAP

STEP-3 117 110 112 122 111 149 151 141 123 147 157 133
l<p l<p l<p l<p l>p Pivot
SWAP

STEP-4 117 110 112 122 111 123 151 141 149 147 157 133
l<p l<p l<p l<p l>p l>p l>p l>p l>p Pivot
SWAP

STEP-5 117 110 112 122 111 123 133 141 149 147 157 151
l<p l<p l<p l<p l>p Pivot SWAP Pivot
SWAP

STEP-6 117 110 112 122 111 123 133 141 149 147 151 157
l>p Pivot l>p Pivot SWAP
SWAP

STEP-7 110 117 112 122 111 123 133 141 147 149 151 157
l<p l>p Pivot l>p Pivot SWAP
SWAP

STEP-8 110 111 112 117 123 133 141 147 149 151 157
Pivot l>p l>p l>p l>p l>p l>p l>p l>p l>p l>p
SWAP

STEP-9 110 111 112 117 122 123 133 141 147 149 151 157
Pivot l>p l>p l>p l>p l>p l>p l>p l>p l>p l>p
SWAP

STEP-10 110 111 112 117 123 123 133 141 147 149 151 157
Pivot l>p l>p l>p l>p l>p l>p l>p l>p l>p l>p

It takes 10 steps to sort the unsorted array using the last element as pivot element.

Program :

```
1  #include <stdio.h>
2  int partitionLast(int a[], int low, int high) {
3      int pivot = a[high];
4      int i = low - 1, j, temp;
5      for (j = low; j < high; j++) {
6          if (a[j] <= pivot) {
7              i++;
8              temp = a[i];
9              a[i] = a[j];
10             a[j] = temp;
11         }
12     }
13     temp = a[i + 1];
14     a[i + 1] = a[high];
15     a[high] = temp;
16
17     return i + 1;
18 }
19 void quickSortLast(int a[], int low, int high) {
20     if (low < high) {
21         int p = partitionLast(a, low, high);
22         quickSortLast(a, low, p - 1);
23         quickSortLast(a, p + 1, high);
24     }
25 }
26 int main() {
27     int a[] = {10, 7, 8, 9, 1, 5};
28     int n = 6, i;
29     quickSortLast(a, 0, n - 1);
30     printf("Sorted array:\n");
31     for (i = 0; i < n; i++)
32         printf("%d ", a[i]);
33     printf("\nCH.SC.U4CSE24108\n");
34     return 0;
35 }
```

Output :

Output

Sorted array:

1 5 7 8 9 10

CH.SC.U4CSE24108

III) Middle Element as Pivot

Program :

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int partitionRandom(int a[], int low, int high) {
4      int random = low + rand() % (high - low + 1);
5      int temp = a[random];
6      a[random] = a[high];
7      a[high] = temp;
8      int pivot = a[high];
9      int i = low - 1, j;
10     for (j = low; j < high; j++) {
11         if (a[j] <= pivot) {
12             i++;
13             temp = a[i];
14             a[i] = a[j];
15             a[j] = temp;
16         }
17     }
18     temp = a[i + 1];
19     a[i + 1] = a[high];
20     a[high] = temp;
21     return i + 1;
22 }
23 void quickSortRandom(int a[], int low, int high) {
24     if (low < high) {
25         int p = partitionRandom(a, low, high);
26         quickSortRandom(a, low, p - 1);
27         quickSortRandom(a, p + 1, high);
28     }
29 }
30 int main() {
31     int a[] = {10, 7, 8, 9, 1, 5};
32     int n = 6, i;
33     quickSortRandom(a, 0, n - 1);
34     printf("Sorted array:\n");
35     for (i = 0; i < n; i++)
36         printf("%d ", a[i]);
37     printf("\nCH.SC.U4CSE24108\n");
38     return 0;
39 }
```

Output :

Output

Sorted array:

1 5 7 8 9 10

CH.SC.U4CSE24108